Entrada e Saída de dados

A Entrada e Saída dos dados permitem que um programa faça interação com o usuário. É possível utilizar o teclado do computador ou notebook para inserir dados a uma variável, ou usar arquivos de disco, entre outras formas.

Para a Entrada de dados, usamos as classes que permitem fazer a leitura dos dados, sendo elas: Scanner, BufferedReader e inputStreamReader.

A Saída de dados trata-se da exibição em tela do valor da variável, gravado na Entrada dos dados. As classes que utilizamos para realizar a Saída dos dados, e que fornecem métodos para envio dos dados são: System.out, PrintWriten e FileOutputStream.

Utilizando as classes de Entrada e Saída de dados, podemos implementar maneiras diferentes de exibir os dados; e até podemos fazer operações matemáticas utilizando a entrada e saída de dados.

Veremos neste conteúdo como podemos fazer uma Entrada e Saída de dados simples, utilizando algumas das classes citadas acima. Tenha o objetivo de entender como a linguagem precisa ser direcionada para realizar alguma operação de entrada ou saída.

* 1. Lembre-se

O processamento de dados é desenvolvido em três etapas, que são: Entrada, Processamento e Saída.

**Processamento**

**Saída**

**Entrada**

Entrada: armazena o dado que o usuário digitar pelo teclado, sendo um Comando de Entrada de Dados.

Processamento: podem ser processamentos de cálculo ou leitura dos dados.

Saída: mostra o resultado do processamento na tela do computador, sendo um Comando de Saída.

1.2 Entrada de dados

Utilizamos a classe Scanner para obter informações do usuário mediante o seu teclado. Esta classe pertence ao Pacote java.util, e precisamos instanciar um objeto desta classe para utilizarmos a Classe.

Veja a sintaxe do Scanner no exemplo abaixo:

Scanner nomeScanner = new Scanner(System.in);

Na aula anterior aprendemos sobre o método main(), e será dentro dele que vamos implementar o Scanner.

Public static void main(String[] args){

// Implementando o Scanner

Scanner leia = new Scanner(System.in);

}

No exemplo acima foi criado um objeto chamado leia, e este objeto pertence a Classe Scanner.

Este objeto foi instanciado com o objetivo de ler o que for inserido no teclado, usando o padrão de entrada (System.in). Na Classe System existe uma variável chamada in, onde a sua função é armazenar o dado que dar entrada no teclado ou em outro arquivo de texto.

Então, vamos resumir o código que temos até o momento, conforme o próximo quadro:

* Temos a importação do pacote java.util.Scanner,
* Temos a nossa Classe EntradaDados,
* Temos o método main definido,
* E temos instanciado o objeto Scanner.

import java.util.Scanner;

public class EntradaDados{

Public static void main(String[] args){

// Implementando o Scanner

Scanner leia = new Scanner(System.in);

}

}

1.3 Método de Entrada da Classe Scanner

Veremos alguns dos métodos de entrada mais usados da classe Scanner.

nextInt(): Faz a leitura de números inteiros.

Int a = leia.nextInt();

nextDouble(): Faz a leitura de números de ponto flutuante de precisão dupla.

double b = leia.nextDouble();

nextFloat(): Faz a leitura de números de ponto flutuante.

float c = leia.nextFloat();

nextLong): Faz a leitura de números inteiro longo.

double d = leia.nextDouble();

nextBoolean(): Faz a leitura de valor booleano.

boolean e = leia.nextBoolean();

next(): Faz a leitura de valores tipo String.

String f = leia.next();

nextLine(): Faz a leitura de uma linha inteira como uma String.

String g = leia.nextLine();

nextByte(): Faz a leitura de valores como byte.

byte h = leia.nextByte();

nextShort(): Faz a leitura de valores como short.

short i = leia.nextShort();

nextBigInteger(): Faz a leitura de valores como BigInteger.

BigInteger j = leia.nextBigInterger();

nextBigDecimal(): Faz a leitura de valores como BigDecimal.

BigDecimal l = leia.nextBigDecimal();

hasNext(): Declara verdadeiro se houver outro valor na entrada.

if (leia.hasNext()){

// Processa o próximo valor

}

hasNextInt(): Retorna verdadeiro se o valor de entrada puder ser interpretado como inteiro.

if (leia.hasNextInt()){

int num = leia.nextInt();

}

hasNextDouble(): Retorna verdadeiro se o próximo valor de entrada puder ser interpretado como um número de ponto flutuante de precisão dupla.

if (leia.hasNextDouble()){

double num = leia.nextDouble();

}

1.4 Exemplo do uso da Classe Scanner

Veja abaixo o exemplo de uso da Classe Scanner implementada com os métodos: nextInt(), nextDouble(), next() e nextLine().

import java.util.Scanner;

public class ScannerExample {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Digite um número inteiro: ");

int numInt = scanner.nextInt();

System.out.print("Digite um número de ponto flutuante: ");

double numDouble = scanner.nextDouble();

System.out.print("Digite uma palavra: ");

String word = scanner.next();

System.out.print("Digite uma linha de texto: ");

scanner.nextLine(); // Consumir a nova linha restante

String line = scanner.nextLine();

System.out.println("Número inteiro: " + numInt);

System.out.println("Número de ponto flutuante: " + numDouble);

System.out.println("Palavra: " + word);

System.out.println("Linha de texto: " + line);

scanner.close();

}

}

Mas o que foi feito no código acima?

Utilizamos os métodos nextInt(), nextDouble(), next() e nextLine() para armazenar os valores que o usuário escreveu via teclado:

nextInt(): realiza a leitura do número inteiro fornecido pelo usuário.

nextDouble(): realiza a leitura de um número ponto flutuante fornecido pelo usuário.

next(): realiza a leitura de uma única palavra fornecida pelo usuário.

nextLine(): realiza a leitura de texto fornecida pelo usuário. Note que o scanner.nextLine() adicional antes da leitura da linha de texto é necessário para consumir a nova linha restante após next() ou nextInt().

Ao utilizar a Classe Scanner, é importante utilizar o comando scanner.close() para terminar de utilizar a classe e liberar os recursos associados.

2.0 Saída de Dados

Bem, em nossa aula observamos vários exemplos de como podemos realizar a entrada de dados, e consequentemente observamos a saída desses dados. Mas ainda não falamos do Método de Saída.

Para os Métodos de Saída, temos: print(), printf() e printIn(). A diante, veremos o que cada um desses métodos faz.

print(): exibe na tela uma cadeia de caracteres (String), sem se preocupar com a formatação.

printf(): Exibe os dados no formate data, hora, moeda, entre outros.

printIn(): semelhante ao método print(), sendo a sua diferença ele pular uma linha depois de exibir o valor.

O exemplo mias simples de Saída de dados é o “Hello World” que realizamos, mas você pode implementar o que será exibido na saída de dados, de acordo com a sua necessidade.

Um recurso que temos para a Saída de dados é a Expressão de Controle, que é uma sequência de caracteres que determina as informações que serão mostradas na tela.

Existem dois tipos de Expressão de Controle: Caracteres comuns, podendo ser um texto qualquer; e Código de controle, que especifica o formato.

Os códigos de controle são utilizados para indicar o tipo de dado que será impresso na tela. São utilizados junto com o sinal “%”, adicionados junto ao argumento. Para cada código de controle, deverá existir uma variável correspondente.

Podemos ver abaixo o código e o tipo de dados:

%c: Caractere simples tipo char;

%s: Cadeia de caractere tipo String;

%d: Inteiro tipo int ou long;

$f: Real com ponto flutuante tipo float ou double;

%.2f: Real com ponto flutuante tipo float ou double;

%e: Real em notação científica com “e” minúsculo tipo float ou double;

%E: Real em notação científica com “E” maiúsculos;

%%: Imprime o próprio caractere “%”.

Veja o código abaixo, como as variáveis são exibidas na saída da tela:

package exibir\_variaveis;

public class ExibirVariaveis {

public static void main(String[] args) {

int quantidade = 1000;

long identificador = 10000l;

float altura = 25.4f;

double area = 45.4567;

double tamanho = 650000000.00;

char tipo = 'A';

String palavra = "Novotec Terceiro Técnico A";

System.out.printf("Variável quantidade = %d", quantidade);

System.out.printf("Variável identificador = %d", identificador);

System.out.printf("Variável altura = %.2f", altura);

System.out.printf("Variável area = %.2f", area);

System.out.printf("Variável tamanho = %e", tamanho);

System.out.printf("Variável tamanho = %E", tamanho);

System.out.printf("Variável tipo = %c", tipo);

System.out.printf("Variável palavra = %s", palavra); } }

Podemos também utilizar os caracteres de escape, que são instruções inseridas no começo ou no final de um texto. Veja abaixo quais são:

\n: vai para a nova linha;

\t: realiza a função que a tecla Tab faz;

\r: move o cursor do terminal para o inicio da linha;

\b: realiza a função da tecla Backspace;

\’: Aspas simples;

\”: Aspas duplas;

\\: Barra invertida.

Observe no código abaixo como podemos utilizar os caracteres de escape:

package exibir\_variaveis;

public class ExibirVariaveis {

public static void main(String[] args) {

int quantidade = 1000;

long identificador = 10000l;

float altura = 25.4f;

double area = 45.4567;

double tamanho = 650000000.00;

char tipo = 'A';

String palavra = "Generation";

System.out.printf("\nVariável quantidade = %d", quantidade);

System.out.printf("\nVariável identificador = %d", identificador);

System.out.printf("\nVariável altura = %.2f", altura);

System.out.printf("\nVariável area = %.2f", area);

System.out.printf("\nVariável tamanho = %e", tamanho);

System.out.printf("\nVariável tamanho = %E", tamanho);

System.out.printf("\nVariável tipo = %c", tipo);

System.out.printf("\nVariável palavra = %s", palavra);

}

}

3.0 Resumindo

Vamos resumir os passos para criar um Scanner:

1. Primeiro Passo: importe a classe java.util;
2. Segundo Passo: crie uma instância da classe Scanner que coletará os dados via console;
3. Terceiro Passo: crie um método para ler os dados (tenha atenção para o tipo de dado que deseja ler);
4. Quarto Passo: feche o Scanner.
5. Quinto Passo: exiba os dados com os métodos de Saída.

Veja no exemplo abaixo como os passos podem ser implementados:

// Primeiro Passo: Importe a Classe java.util

Import java.util.Scanner;

Public class Scanner{

Public static void main(String[] args) {

// Segundo passo: Crie um objeto Scanner para ler a entrada

Scanner leia = new Scanner(System.in);

// Terceiro Passo: crie um método para ler os dados

System.out.print(“Informe um número inteiro: ”);

int valorInt = leia.nextInt();

System.out.print(“Informe um número float: ”);

float valorFloat = leia.nextFloat();

scanner.nextLine(); // Consumindo a nova linha

System.out.print(“Informe um número string: ”);

String valorString = leia.nextLine();

// Quarto Passo: Fechando o scanner

leia.close();

// Quinto Passo: Exibindo os valores lidos

System.out.print(“Número inteiro: ” + valorInt);

System.out.print(“Número float: ” + valorFloat);

System.out.print(“String: ” + valorString);

}

}

Conclusão

E aí, como se saiu? Parece ser complicado aprender tudo isso, mas a prática sempre te ajudará a aprender, e com o tempo você notará que não é tão difícil assim.

Não tenha pressa de aprender, vá com calma! Entender como essa lógica funciona não é tão simples assim, mas o que você precisa fazer é continuar estudando, mesmo que não esteja fácil.

Ah, quase me esqueci da última dica: não se compare com outros. As vezes é normal uma pessoa ter uma rápida compreensão de um conteúdo, algo que talvez você não tenha. Isso não quer dizer que você não é capaz de fazer as coisas, só vai levar um tempo a mais para entender o mesmo conteúdo que outra pessoa entendeu mais rápido.

Se você compreender um assunto, ajude o colega que ainda não entendeu a matéria, isso ajudará no andamento do conteúdo, no seu aprendizado e no aprendizado do colega.

Referências

[Drops] Java – I/O: Entrada e Saída de dados

https://www.linkedin.com/pulse/drops-java-io-entrada-e-sa%C3%ADda-gabriel-d-emery/

Java: Como ler e apresentar dados com os comandos de entrada e saída

https://www.devmedia.com.br/java-como-ler-e-apresentar-dados-com-os-comandos-de-entrada-e-saida/22266

Entrada de Dados: Classe Scanner

https://www.devmedia.com.br/entrada-de-dados-classe-scanner/21366

Entrada e Saída de dados

https://github.com/Leon4rdoalves/CookBook-Java/blob/main/04.md